Web Platform Providing Information and Simulation Game of Baseball Metrics

Chaewon Ko, Changwoo Shim, and Hyunchang Shin

Sungkyunkwan University, 2066, Seobu-ro, Jangan-gu, Suwon-si, Gyeonggi-do, Republic of Korea

Abstract. Scientific approaches to interpreting baseball records have yielded diverse metrics for players, ranging from simple to more complex. Recognizing the desire among baseball fans to comprehend these metrics for a deeper enjoyment of the sport, our team came to create a web platform. This platform consists of a record page with detailed metrics descriptions, a simulation game enabling direct assembly of batting orders and real-time result observation, and a page for comparing and analyzing players' abilities through these metrics. Designed on Figma, the web platform is implemented using React.js and FastAPI, respectively. For the game and analysis we applied mathematical and statistical methods such as weighted averages, maximum-minimum normalization, and logistic functions. Through this user-friendly platform, baseball enthusiasts can gain a foundational understanding of baseball metrics and intuitively grasp their impact on the actual games and players' performances.

Keywords: Baseball · Metrics · Web Platform · Simulation Game

1 Introduction

Baseball holds the title of the most popular sport in Korea, boasting the highest average number of spectators per game among professional sports. [1] The sport's popularity stems not only from its entertaining aspect but also from its ambivalent characteristic, evolving into a field of study. Baseball, often referred to as a *sport of record*, involves diverse and intricate approaches to measuring game results and player performance.

Enthusiasts, ranging from casual fans to baseball officials and academics specializing in baseball-related mathematics and statistics, share a interest in evaluating players and predicting the outcome of games. A survey conducted among baseball fans done by ourselves revealed a prevalent desire for a deeper understanding of baseball metrics. Focusing on these metrics and the increasing need to analyze games and players through them, our initiative aimed to create a website offering information on baseball metrics, simulation game, and analytical tools.

The website's development was structured into design, front-end, and back-end components, utilizing the React.js framework for the frontend and FastAPI

for the backend. Its key sections consist of a homepage for initial interaction, a record page for exploring baseball players' statistics, a game page, and an analysis page for comparing and evaluating players' performance metrics. Data necessary for recording purposes was gathered through statistical sources, and the implementation of the game and player analysis systems includes mathematical and statistical methods such as weighted averages, maximum-minimum normalization, and logistic functions. This approach enabled us to create a simulation game resembling an actual baseball match, providing users with a natural understanding of how complicated player metrics influence games and performance.

2 Design

2.1 Overall Architecture

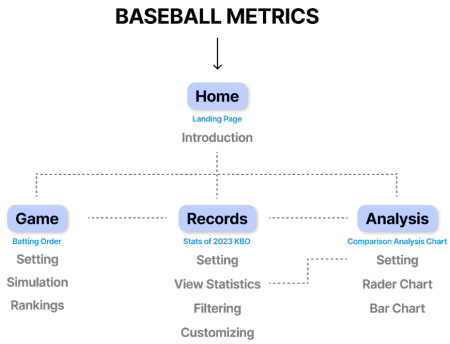

Fig. 1. Information Architecture of *Baseball Metrics*

To engage baseball fans and cultivate their interest in baseball indicators, our website is thoughtfully designed with four distinct pages. Figure 1is the overall structure of our website, name *Baseball Metrics*. The homepage serves as an introduction to the website, offering a brief overview and a navigation button to access the main functions. Additionally, the record page provides a comprehensive look at baseball players' statistics, along with concise descriptions of each record.

The main function of our website is located in the game and analysis pages. On the game page, users can immerse themselves in the world of KBO players' records, strategically forming an actual baseball batting order. By selecting players, arranging the batting order, and incorporating pinch hitters, users can run simulations. To enhance understanding, we provide *indicator utilization tips* tailored to each player's at-bat status and the game situation. This feature should offer insights into the significance of each player's index in the context of a baseball game.

The analysis page enables users to have a deep-dive into player metrics, presenting a visual representation of a specific player's *ability value* compared to a designated benchmark. This *ability value* is a proprietary index that is calculated from actual recorded values corresponding to various indicators—an abstract yet meaningful representation of complex and raw metrics data. Through intuitive graphs, users can compare and analyze a player's performance in relation to specific indicators, facilitating an objective and comprehensive analysis of player records.

2.2 System Architecture

Overall system architecture is as shown as Figure 2.

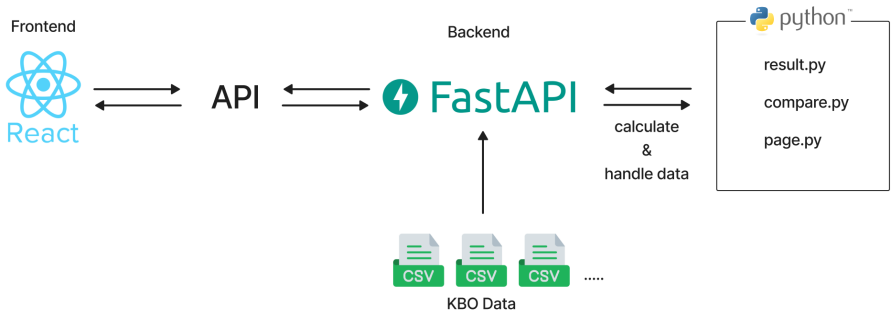

Fig. 2. System Architecture

Description of our main algorithms are as follows.

Simulation Game

1. When a pitcher is selected, send the *name* of the chosen pitcher to the backend.
2. Upon selecting a batter, deliver the *names* of all 15 chosen batters to the backend. The first nine batters are from the starting lineup, and the subsequent six batters (10th to 15th) serve as substitutes.
3. Pressing the progress button triggers the backend to set the pitcher and current batter based on the received information. The backend executes the function and transmits at-bat results, base situations, outs, and inning scores to the frontend.
4. Render the received data from the frontend.
5. Pressing the Replace button forwards the *Name* of the replacement player to the backend.
6. If the bunt button is pressed, execute the bunt function to obtain at-bat results, base situations, outs, and scores for each inning.

7. Post-game, inquire about registration in the ranking. Receive the desired name for ranking entry, and upon pressing the save button, transmit the pitcher's name, starting lineup arrangement, total score, and number of home run hits to the backend for storage.
8. Selecting a pitcher on the game ranking page transmits the "pitcher name" to the backend. The backend sends ranking data for the pitcher to the frontend, which then renders the received data on the ranking page.

Player Analysis Page

1. Choose two players and press the Compare button to send the player names *Name 1* and *Name 2* to the backend.
2. Forward the capabilities and indicators corresponding to the names from the pre-written Abill.csv.
3. Render the graph using Radar and Bar components provided by React-chartjs-2, incorporating the corresponding values at the frontend.

2.3 Core Skill and Technique

Data We used the Papa Parase, CSV Parser for JavaScript library for managing csv files.

Weighted Average A calculated average that incorporates the weight assigned to each data value based on its importance or influence in the overall data set. It is introduced to holistically assess one's competency by considering multiple indicators simultaneously. The determination of each weight is rooted in the subjective judgment of the team members.

Maximum-Minimum Normalization A normalization technique that confines all data within the [0,1] range. It is applied to establish competency values ranging from 0 to 100 during self-assessment of competency.

Logistic Function A mathematical function characterized by a sigmoid curve, ensuring it is both bounded and differentiable. It features a single inflection point with non-negative differential values across all points. It is introduced to simultaneously consider individual players' indicators and their opponents' indicators. The application becomes crucial as it accommodates varying impacts of relative indicators based on the number of at-bats. Before the inflection point, relative indicators have a limited effect, but post-inflection, they significantly influence the overall assessment of the player's performance.

2.4 Challenges

Providing Description of Metrics Addressing the challenge of presenting users with concise versions of baseball indicators, we encountered the complexity of vast and varied information sources. To ensure objectivity and consistency in our descriptions, we opted to reference the MLB's official league website. This decision was grounded in the recognition of MLB as the most renowned and prestigious professional baseball league globally, ensuring the highest reliability of information.

To maintain uniformity in our explanations, we adopted a standardized sentence structure, stating that each indicator is based on [recorded values] and holds a [relation] to [athlete's performance]. This approach aims to provide users with clear and consistent information, minimizing the risk of subjective or inconsistent interpretations in the process of summarizing the expansive details of baseball indicators.

Creating a Metrics Record Table To address the challenge of importing new tables for player selection or fetching player indicator tables on the record page, we implemented a solution involving the generation and transmission of CSV files.

For the pre-game player selection page, where the index used in the game is fixed, information such as team, reference index, and page is sent to resolve the issue. Sending this data enables the corresponding CSV file to be created and fetched.

On the record page, where records can be categorized and new tables need to be generated, we provide the functionality to customize indicators. When a category is selected, an array of pre-configured "indexes" or a customized "indexes" array is sent. The backend reconstructs the corresponding CSV file, which is then rendered at the frontend. This approach allows us to render large CSV files, including over 300 indicators (including duplicates) for 285 pitchers and 293 batters. It efficiently handles about 20 indexes of 30 players needed at that specific moment.

Data Processing In the process of crawling the Statiz site, loading all the data at once proved challenging. Consequently, we adopted a page-by-page crawling approach, resulting in the creation of 26 CSV files. To streamline the data, we consolidated 14 pitcher indicators into a single file named PitchAll.csv and combined 12 batter indicators into another file called BatteriesAll.csv.

For player selection in the game, we identified the top pitchers in the league and hitters with a certain number of games played. Extracting this information from the comprehensive All.csv file, we promptly generated the response.csv file on the backend and transmitted it to the frontend. This approach ensures an efficient and targeted selection of players for use in the game, enhancing the overall user experience.

3 Implementation

3.1 UX/UI

The website's top navigation features dedicated buttons for seamless navigation to each page, offering users the flexibility to access their desired content effortlessly. For metrics descriptions, recognizing the importance of viewing with the players' records, we implemented an intuitive solution. As shown in the Figure 3, the description window is popped up through a mouse hover, allowing users to grasp information at a glance without disrupting their current page.

Fig. 3. Enter Caption

The game simulation screen is inspired from the scoreboard and relay board of a baseball game, aiming to enhance understanding while minimizing fatigue caused by complex visuals. If you look at the Figure 4, this screen provides a familiar depiction of the game's progress, allowing users to check the score and game advancements for each inning in a manner similar to what baseball fans are accustomed to.

For player comparison analysis, we prioritize user-friendly experiences. Instead of long descriptions or intricate numerical data, we opted for a visual approach. Graphs displayed on the right interface in Figure4 are thoughtfully incorporated to facilitate a clear and accessible understanding of the analysis, ensuring a user-friendly and engaging exploration of player comparisons.

3.2 Frontend

In the player comparison feature, we've integrated radar and bar graphs using the react-chartjs-2 library. To manage page navigation, we leverage the

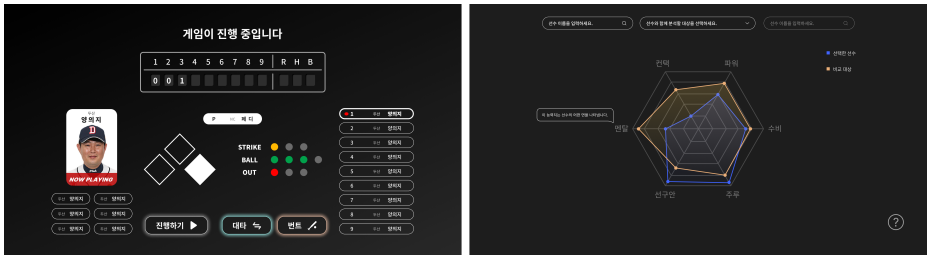

Fig. 4. Simulation Game and Visual Analysis Graph Interfaces

BrowseRouter, Routes, and Link components from the react-router-dom package. For consistent theming across the application, we utilize the ThemeProvider from styled-components.

Given the unique fan culture in the KBO, where themes can be aligned with specific team colors, we've implemented a dynamic theming system. Themes.js defines mainTransparent and subTransparent colors for each club, with mainColor, subColor, and transparency set to 0.2. The selected theme in Header.js is passed to App.js via onThemeChange=handleTheme, utilizing a callback function to handle value transfer from child to parent components.

App.js, acting as the parent component, forwards the currentTheme value to its subcomponents. The theme is then applied in the subcomponents using styled-components' styled functionality. This comprehensive approach ensures a cohesive and customizable theming experience throughout the application, reflecting the vibrant nature of KBO fan culture.

3.3 Backend

For Backend, we used FastAPI and python language to create algorithms for simulation game and ability calculation. FastAPI is a web framework for building APIs for Python 3.6+ which has a modern, fast and high performance. The URL Pattern, HTTP Method (POST/GET), function name, and Description of the API created for communication with the frontend are as shown in the Figure 5.

4 Evaluation

4.1 Method

An evaluation was carried out to assess the resemblance between our self-produced game algorithm and the actual game flow. In practice, the simulation replicates the lineup from historical baseball games—the batting order for a specific pitcher—and iterates through the process 20 times. The aim is to determine the validity of the game operation by comparing the average values of these 20 iterations with the actual results.

API

	A	B	C	D	E
1	**URL**	**Method**	**Function**	**Request Body**	**Description**
2	/selectplayer	POST	select_player	{“name” : str}	Get batter’s name and append to ordered player array (starting player)
3	/selectpinch	POST	select_pinch	{“name” : str}	Get pinched hitter’s name and append to pinched player array
4	/selectpitcher	POST	select_pitcher	{“name” : str}	Get pitchers name and set game’s pitcher
5	/play	GET	play		Give result of PA to frontend.
6	/bunt	GET	bunt		Give result of PA to frontend.(when user click bunt button)

7	/page	POST	page	{
				“page” : int,
				“sort” : str
				}
					Get page index and sort column.
					Make csv file by input data.
					Then transmit csv file to frontend.

8	/teampage	POST	teampage	{“team” : str}	Get team name. Filtering player and make csv file. Then transmit to frontend

9	/compare	POST	compare	{
				“type” : str,
				“player1” : str,
				“player2” : str
				}
					Transmit data which is necessary for comparing two player to frontend.

10	/change	POST	change	{
				“name1” : str,
				“name2” : str
				}
					Change player between starting hitter(name1) and pinched hitter(name2)

Fig. 5. API Used to Communicate with Frontend

4.2 Results

Table 1. Evaluation Result: Simulation Result Compared to Real Game Result

	삼성 vs 반즈	NC vs 엘리아스	한화 vs 알칸타라
Simulation Result	1 point, 3 hits, 3 walks	1.05 home runs, 13.85 hits, 1.70 walks	10.95 hits, 2.40 walks, 3.25 runs
Expected Result	1.5 runs, 4.5 hits, 4.5 walks	12 runs, 24 hits, 3 home runs	3.9 hits, 2.6 walks, 0 runs

Three actual matches were selected for comparison for evaluation, and the information used in the match is as follows.

Samsung Lions batters and pitcher Barnes (2023.10.03 game) Real Game Results: (6 innings), one run, three hits, three walks, and zero home runs were allowed. [3]

NC batters and pitcher Elías (2023.10.03 game) Real Game Results: (3 innings) Quick Hook (fast replacement of starters) One trick, eight hits, one home run, and four runs. [4]

Hanwha batters and pitcher Alcántara (2023.05.02 game) Real Game Results: (7 innings), three hits, two walks, and zero runs were allowed. [5]

Here, the number of innings in each game is converted to fit the number of innings in the simulation game, 9 innings, and the result of comparing it with the average value of the 20 game results is shown in the Table 1.

Upon comparison, the game consistently tends to achieve higher scores. This tendency arises from the fact that batter performance is not only gauged against proficient pitchers but also against those with comparatively lesser proficiency. In an effort to rectify this, adjustments were implemented through logistic functions, though these adjustments appear to be somewhat insufficient. Nevertheless, the system performs significantly better than when a simple sigmoid function was in place.

Moreover, the selection of games from the early part of the season as the benchmark for actual results yielded better-than-expected outcomes. Given the extended duration of the season, variations in performance are inevitable, and the games chosen for testing may not necessarily represent the player's average performance.

5 Limitations and Discussions

5.1 Assumptions

Despite the vast array of over 100 indicators utilized in real baseball, our game focuses on a streamlined selection of 10 pitchers and 10 batters metrics each.

For pitcher selection, we include Win-Loss record, Innings per Game(total innings/entrance), Earned Run, Hit, Home Run, Strikeout, Walk, ERA, WHIP, and Pitch Type. For batters, the chosen metrics include AVG(Batting Average), BB(Walks), SO(Strikeouts), GDP(Ground into Double Play), SLG(Slugging Percentage), OBP(On-base Percentage), BA/RISP(Batting Average with Runners in Scoring Position), phLI(Pitcher's Hitting Influence), GO/AO(Ground Ball/Fly Ball Ratio), and BB/K(Walks to Strikeouts Ratio).

For implementation and better user experience, the game excludes the existence of an opposing team. In this simplified setting, unlike real baseball, it is assumed that there is only one pitcher for our team's batters during the game. Additionally, the user can choose between only two actions—*bunt* and *pinch-hitte*r—during an at-bat.

5.2 Limitations

Gap Between Real Baseball and Simulation Incorporating every facet of baseball proved challenging for our game development. Actual baseball games involve a myriad of situations and strategies, including stealing, bunting, balks, defensive errors, defensive shifts, and more. The dynamics of a baseball game shift accordingly based on these elements. However, translating these complexities into a realistic development proved to be a formidable task, and we faced limitations in fully reflecting all these nuanced aspects within our simulation.

Reliability of Self-Supplied Information We developed the simulation game algorithm independently, leveraging a solid understanding of baseball rules. Our aim was to refine it, ensuring that the gameplay reflects an understandable flow to ordinary baseball fans through regularization. However, as previously mentioned, the game wasn't crafted with consideration for every nuance of a baseball game. It relies on our assumptions, which were set up by subjective judgments. Consequently, it's challenging to assert that the process and outcome of the game maintain complete objectivity.

Furthermore, while the formula for calculating a player's ability value utilizes actual metrics, the selection of specific metrics involves subjective judgment. Additionally, there are limitations in considering certain factors in calculating each ability value due to constraints in actual statistical data. Take the example of the *home run* indicator—variables such as the stadium dimensions and the height of the home run fence, along with daily wind conditions, contribute to the variability of home run outcomes. In essence, the home run indicator can be influenced by luck, introducing a limitation where the reliability of the provided information may be affected by these statistical loopholes.

6 Related Work

Our work is concentrating on a platform that helps users understand baseball metrics in a user-friendly way, and there are no specific work with similar intentions. Yet there are some works that have some similar features as follows.

6.1 Domestic Web Platforms

KBO Website, Statiz 'KBO' and 'Statiz' can be better in providing extensive baseball game and player records. However, they fall short in offering concise and abbreviated information - they only provide Korean words for metrics names that are written in English abbreviations. In contrast, our website focuses on the most recent season data but stands out by delivering a succinct yet comprehensive description of metrics along with the records. Additionally, it also provides a function to select only the desired metrics or to view it by designating a range of values of the metrics. This approach aims to strike a balance between detailed information and user-friendly accessibility, addressing a gap observed in other platforms.

6.2 Simulation Game

OOTP(Out of The Park Baseball) One of the representative baseball simulation games is OOTP. This sophisticated simulation seamlessly incorporates historical league play, fluctuations in salary prices, league expansion, team dynamics, and statistical figures, creating an immersive experience that closely mirrors real-life baseball leagues [2]. However, its complexity and extensive features may pose a significant learning curve and inconvenience for users. Furthermore, OOTP is a paid game with installations on devices. In contrast, our game system operates on a web platform, providing free accessibility to anyone with an internet connection. The game requires only basic settings—opposite batters and batting order—allowing users to play without the need for extensive background knowledge. The remarkably simple and intuitive user interface further minimizes the user's burden.

Table 2 summarizes the differences in our work *Baseball Metrics* compared to the related works mentioned above.

Table 2. Position of *Baseball Metrics* Compared to Prior Works

	Vast Database	**Brief Description of Metrics?**	**Easily Accessible**	**Visual Analysis of Player?**	**Readable & Intuitive UI**
KBO	O	X	O	X	O
Statiz	O	X	O	X	X
OOTP	O	X	X	O	X
Baseball Metrics	X	O	O	O	O

7 Conclusion

In this report, we clarified the motivation and objectives behind the creation of this platform, providing a detailed description of the design and the actual

development process. Although there are some imperfections in the realism of the algorithm within the game and analysis, the core functionalities of the system, and we recognize the limitation in dealing with a more constrained dataset compared to existing web platforms, this project holds immense significance. It represents a pioneering effort to fill the gap in baseball understanding and metrics. For those who are interested in baseball metrics but frustrated by the complexity, this platform will work as a comfortable and enjoyable solution. It also serves as a promising gateway, inviting those unfamiliar with baseball metrics into a new and intriguing field.

References

1. 문화체육관광부. (2023). 프로스포츠 경기단체 자체 통계자료. *주요 프로스포츠 경기 수 및 경기당 평균 관중 수*.
2. 2023 OUT OF THE PARK DEVELOPMENTS, https://www.ootpdevelopments.com/out-of-the-park-baseball-korean/. Last access 9 Dec 2023
3. 네이버 스포츠, https://m.sports.naver.com/game/20231003SSLT02023. Last access 10 Dec 2023
4. 네이버 스포츠, https://m.sports.naver.com/game/20231003NCSK02023/record. Last access 10 Dec 2023
5. 네이버 스포츠, https://m.sports.naver.com/game/20230502HHOB02023/record. Last access 10 Dec 2023